ISOM 671: Managing Big Data
(MID-TERM LEARNING AMPLIFICATION EXAM)

Your Name
Your Email

There are 6 numbered questions spread over 7-pages. Before you begin, please refer to the notes at the end of this document. **Please submit your responses as a single PDF file by uploading it to the course canvas page.** ***(Note: You have 12-hours to complete this take-home exam and it has a late penalty of 25% for every 6-hour delay.)***

1. (9 points) Answer the following short questions:

1.1. Briefly describe the requirements of a 3rd normal form (3NF).
1.2. Briefly describe M:N relationship between two entities (share an ER model)
1.3. Briefly describe when "WHERE" clause can and cannot be used to select rows after a "GROUP BY" clause?
1.4. Briefly discuss three differences between normalized modeling (OLTP database) and dimensional modeling (OLAP database).
1.5. Briefly discuss two differences between the NoSQL document database and a NoSQL wide column database.
1.6. Briefly discuss what the following command returns when used on documentDB: db.restaurants.find({"address.location": location})
1.7. Briefly discuss the difference between "scan" and "get" in HBase
1.8. Briefly discuss why NoSQL databases are not good for "join" and "group" operations?
1.9. Briefly discuss the max size of HRegion you will select when designing a database for TikTok?

2. (4 points) Write SQL queries for the following cases (using sakila database):

2.1. Check if a movie is in stock: list inventory_ids that are currently in stock (across all locations) for movie "ACADEMY DINOSAUR"
2.2. For movie in stock, in the customer location, checkout the movie (in rental table) for customer = "DWAYNE OLVERA" (assume your staff_id = 1).
2.3. For rental entered, collect the rental payment for the movie and add a row in the payment table.
2.4. Create a list of overdue movies (i.e., rental_duration < '2006-02-18' – rental_date)

3. (6 points) Write SQL functions to calculate the sentiment from text (using Yelp review data):

3.1. Load the positive and negative words data[1,2] using "LOAD DATA LOCAL INFILE" into two separate tables: pos_words and neg_words.
3.2. Write mySQL user-defined-functions that takes a text input and returns the number of positive and negative words (that exist in pos_words and neg_words tables) in that text.

[1] https://raw.githubusercontent.com/ihumanoid/analytics/main/data/negative-words.txt
[2] https://raw.githubusercontent.com/ihumanoid/analytics/main/data/positive-words.txt

3.3. Load the Yelp review data[3] (from previous homework assignment) and write a SQL query to show top 5 businesses with highest average positive words and highest average negative words. You can test your code on only 1000 rows of data. (*Hint: for simplicity, use LIMIT with UNION*)

3.4. Using the result of previous query as a subquery, write a query to show the sentiment of each review, where sentiment is calculated as: ("number of positive words" - "number of negative words")/(total words). (*For simplicity, you can use total number of positive and negative words as total words*)

4. (7 points) Write SQL queries for the following cases (database table schema provided):

4.1. (1 point) Random row sample

`big_table`

column	type
id	int
name	varchar

Let's say we have a table with an *id* and *name* field. The table holds over 100 million rows, and we want to sample a random row in the table without throttling the database.

Write a query to randomly sample 5 rows from this table

HINT: mySQL has a rand() function (https://dev.mysql.com/doc/refman/8.0/en/mathematical-functions.html#function_rand)

4.2. Employee Salaries (ETL error)

employees table (representing a company payroll schema)

column	type
id	integer
first_name	string
last_name	string
salary	integer
department_id	integer

[3] https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset/versions/6

Due to an ETL error, annual compensation adjustments (salary updates) were **inserted** as a new row in the employee table (vs updating the salaries in the existing employee row). The head of HR still needs the current salary of each employee.

Write a query to get the current salary for each employee.
Assume no duplicate combination of first and last names. (i.e., No two John Smiths)

HINT: every time a new row is inserted, the id is autoincremented.

Output Table

column	**types**
first_name	string
last_name	string
salary	integer

4.3. Monthly Customer Report

`transactions` table

column	**type**
id	integer
user_id	integer
created_at	datetime
product_id	integer
quantity	integer

`products` table

column	**type**
id	integer
name	string
price	float

`users` table

column	type
id	integer
name	varchar
sex	varchar

Write a query to show the number of users, number of transactions placed, and total order amount per month

Example Output:

month	num_customers	num_orders	order_amt
2020-01-01	300	550	12000
2020-02-01	315	700	9000
2020-03-01	290	900	20000

4.4. Closest SAT Score

scores table

column	type
id	integer
student	varchar
score	integer

Given a table of students and their SAT test scores, write a query to return the two students with the closest test scores with the score difference.

If there are multiple students with the same minimum score difference, select the student id that is lower.

Example:

Input

id	student	score
1	Jack	1700
2	Alice	2010
3	Miles	2200

id	student	score
4	Scott	2100

Output

one_student	other_student	score_diff
Alice	Scott	90

4.5. Product Recommendation (co-purchases)

`transactions` table

column	type
id	integer
user_id	integer
created_at	datetime
product_id	integer
quantity	integer

`products` table

column	type
id	integer
name	string
price	float

Let's say we have two tables, *`transactions`* and *`products`*. Hypothetically the *`transactions`* table consists of over a billion rows of purchases bought by users. We are trying to find paired products that are often purchased together by the same user, such as wine and bottle openers, chips and beer, etc.

Write a query to find the top 100 paired products and their names.

Output:

column	type
P1	string

column	type
P2	string
count	integer

5. (6 points) Designing Database:

The athletics department at Emory University needs to create a database to track the games and ticket sales across all college sports. There are many college sports, including men's basketball, women's basketball, track and field, swimming, etc. Each college sport has a corresponding university team. For each of these sports (teams), Emory maintains some basic information including team name, description, head coach, and year of establishment. There are also several facilities managed by Emory for these college sports, such as WoodPEC gymnasium, Cooper field, and Brown aquatic center. Each facility has a name, capacity, date of construction, location, and date of last inspection. Each game is between an Emory University team and another college team (e.g., Northwestern, Duke, Purdue). Emory maintains attendance of each game (i.e., number of people who attended the game), game date, final scores for the two teams, whether it is a home game, the facility used for the game, and type of the game (e.g., a conference or exhibition game).

Customers can buy season tickets, or tickets for specific games. Season ticket prices are fixed for each sport. Each customer may buy one or more season tickets per sport. Emory records season ticket sales in terms of who, when, which sport/team, and how many tickets have been purchased. Similarly, Emory also tracks who bought game tickets, how many, for which game, and when. Unlike season tickets, however, game ticket prices may vary from transaction to transaction. Hence, Emory also tracks how much a customer paid per ticket. Emory also keeps track of all customers who bought tickets, including their first name, last name, address (street_address, city, state, zip), email, and telephone.

5.1. (3 points) Database Design (ER Model)

Based on the above description, create an ER diagram. The requirements for the E-R diagram are as follows:

- The design should be in the third normal form (3NF).
- The design should clearly identify all the primary and foreign keys.
- The design should clearly identify all the relationships. Specifically, the cardinality of the relationships should be clearly identified. No explicit many-to-many relationships are allowed.

Create tables and constraints to reverse engineer (draw) your ER diagram and clearly list all your assumptions.

5.2. (3 points) Dimensional Modeling

The next step is to design a data warehouse for game ticket sales (you can exclude season ticket sales). The analytics team at Emory University hopes to conduct multi-dimensional analysis on game ticket sales to answer ad hoc questions such as:

- Which opponent generates the highest ticket sales for a particular sport?
- How far in advance do customers buy tickets for a particular game?
- How are sports ranked in terms of total game ticket sales?
- How do ticket sales vary by year, season, month, week, day of week, facility, customer city, zip code?
- How are sports ranked in terms of total ticket sales per game?

Design a dimensional model with a single fact table (star schema) using MySQL workbench. Briefly discuss what is the grain of your fact table?

6. **(8 points) NoSQL Databases:** This week, Microsoft announced the integration of Dall-E with Bing Chat[4,5]. This product will help generate revenue for both companies Microsoft and OpenAI through paid services, increased ad revenue, and increased market size. Assuming you are now tasked to develop the database system to store and analyze data for this integrated solution, briefly answer the following questions:

6.1. Assuming Bing uses ChatGPT and Dall-E API calls, they don't need to store any of the generative AI models. But they want to store user questions and responses (text/images sent to users), timestamps, and clicks on any link from Bing search engine (assume Bing search data is already stored in wide-column database). Briefly discuss what NoSQL database(s) Microsoft will need to create to store user data from Bing chat (including Bing image and Bing search). You should specify all keys, attributes, and possible values for the proposed design.

6.2. Briefly discuss what two CAP properties this data will need to support.

6.3. Microsoft also decided to use OpenAI APIs for LinkedIn[6]. Knowing that LinkedIn currently uses a Key-Value NoSQL database Voldermort[7], briefly discuss how you would store new data generated by LinkedIn.

6.4. Assuming you are now tasked to perform analytics for Microsoft's use of OpenAI features. More specifically, you want to know what OpenAI features are used at what time of the day/week and from what location. To answer this, you need to build a data warehouse and ETL data from databases you developed for Bing and LinkedIn. Briefly discuss the design of the data warehouse (in star-schema), and provide the fact and dim tables.

[4] https://blogs.bing.com/search/october-2023/DALL-E-3-now-available-in-Bing-Chat-and-Bing-com-create-for-free
[5] https://www.theverge.com/2023/10/3/23901963/bing-chat-dall-e-3-openai-image-generator
[6] https://techcrunch.com/2023/10/03/linkedin-goes-big-on-new-ai-tools-for-learning-recruitment-marketing-and-sales-powered-by-openai/
[7] https://www.project-voldemort.com/voldemort/

NOTES:

For Q1 (easy), limit your answer to a total of 3-pages (double spaced).

For Q2 (easy), you should provide SQL statements, and results of any SQL statement (less than first 5 rows).

For Q3 (medium), you should provide SQL statements, and results of any SQL statement (less than first 5 rows).

For Q4 (hard), please note that this question has some parts with high TTPP (time taken per point); it is intentional! Please optimize your time. Consider the hints provided in the questions.

For Q5 (medium), you should provide ER diagrams only (no need for SQL queries). You don't need to add any data in the tables either.

For Q6 (hard), please limit the answer to 2-page double-spaced text with up to 1 additional page for exhibits (tables, models, etc.). Please note that 80% of the score will be allocated to your analysis, critical reasoning, and writing, and only 20% of the score will be allocated to the "answer"

Fine-print (approximately 286 minutes):
*Easy: less time taken per point (about 3-5 minutes/point) – 13*4 = 52 minutes*
*Medium: More time taken per point (about 6-8 minutes/point) – 12*7 = 84 minutes*
*Hard: Most time taken per point (about 9-11 minutes/point) – 15*10 = 150 minutes*